International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 03 | March - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Autonomous Navigation and Object Detection Using ROS 2 and YOLOv8

Yash Bhaskar[1], Kartik Dolaskar[2], Harshal Jadhav[3], Mahesh Shirke[4] , Prof. Kedar Kulkarni[5]

ybbhaskar19@gmail.com, kartikdolaskar.1106@gmail.com, harshaljadhav6565@gmail.com, maheshshirke1921@gmail.com, kedar.kulkarni@zealeducation.com

[1,2,3,4] Undergraduate Student, Department of Robotics and Automation, Zeal College of Engineering and Research, Pune(MH), India
[5] Assistant Professor, Department of Robotics and Automation, Zeal College of Engineering and Research, Pune(MH), India

---

Abstract - Surveillance is a critical component of national security. Design and creation of a prototype Autonomous Mobile Robot (AMR) for surveillance using a YOLO v8 model consist of innovative aspects for quick surveillance operations across different terrain. The AMR with camouflage-like features to blend in with the environment uses ROS for autonomous navigation and YOLO for precise object detection. camera, face recognition, and motion detection with OpenCV, the AMR also incorporates an alarm notification system. The highly advanced obstacle detection and navigation guiding of the AMR. Field trials demonstrated that the AMR was able to move around without assistance while executing surveillance operations with ease, which demonstrated its prospects as an invaluable tool for use in the military and security services. The project introduces dramatic advancements in autonomous surveillance, with constant real-time monitoring involving limited human intervention. Its pairing with advanced detection systems and long-lasting hardware optimizes its performance in operations, rendering it a suitable option for hostile surveillance.

Key Words: AMR, Surveillance, Open-Cv, YOLO v8, ROS2, Real Time Detection, Google Cloud, Twilio.

1. INTRODUCTION

The application of autonomous systems in surveillance and military use has experienced monumental growth in recent years. Advances in modern robotics and artificial intelligence (AI) have revolutionized how surveillance operations are carried out, with improvements in efficiency and protection of human life. Advanced navigation, detection, and communication features in autonomous mobile robots have made them a key tool across numerous fields, including border patrol, reconnaissance, and disaster response. These robots give unmatched situational awareness and mitigate operational difficulties in high-risk and sensitive environments.

One of the key features of autonomous surveillance systems is that they can be deployed covertly but with real-time monitoring capabilities. Current solutions are mainly centered around particular functionalities such as navigation or detection but fail to provide an end-to-end approach combining stealth, autonomy, and smart detection. Especially in military missions, where stealth and versatility to natural environments are key, there is a requirement for systems that integrate with their environment but provide guaranteed surveillance. Addressing these challenges can significantly enhance the operational capabilities of such systems.

Even with these advances, natural camouflage integration with autonomous navigation and intelligent detection has yet to be fully explored. Most research has centered on sensor accuracy improvement, path planning, or AI-based detection algorithms in a vacuum, rather than implementing all these technologies within a single, durable platform specifically intended for stealth military surveillance. A system that could move on its own, sense movement, and provide real-time notifications and yet be practically invisible to opponents would be a revolutionary leap forward in the discipline.

The goal of this research is to conceptualize and create an autonomous surveillance robot with a stone-mimicking camouflage design, ROS2 for locomotion, and YOLOv8 for object detection. It is hypothesized that the robot will be able to conduct effective covert surveillance in military operations by camouflaging itself to become undetectable and delivering precise real-time warnings. The method includes designing a strong hardware-software integration platform, such as a mild steel chassis, LiDAR sensors, and an alert system. The result of this study will advance the general area of surveillance robotics by proving the viability of combining camouflage with state-of-the-art AI and navigation technologies to increase operational stealth and efficiency.

2. RELATED WORK AND OUR SYSTEM

2.1 Related Work

The recent developments in the area of autonomous surveillance robots have made it possible for robots to effectively move and observe different environments, such as military areas, industrial plants, and distant locations. Various navigation solutions have been proposed to handle issues like real-time obstacle avoidance, environmental mapping, and dynamic path planning.

Integration of vision systems enabled by AI has transformed robotic observation by enabling instant detection of objects and humans in real-time. Older techniques made use of 2D LiDAR obstacle detection. Recently, cost-effectiveness of 3D depth cameras and LiDAR scanners has enhanced robustness and precision of robotic travel significantly. New technologies such as sparse multi-beam LiDAR and stereo vision have eliminated older raycasting methods, especially where objects have high mobility as their positions continuously change.

The Robot Operating System (ROS2) has become a pervasive framework for robot control and navigation. ROS1 was used extensively for the navigation of mobile robots, but with the release of ROS2 as a more scalable and efficient alternative, Navigation2 has gained favor. Navigation2 combines behavior trees for enhanced task coordination and hosts state-of-the-art mapping and localization algorithms optimized for real-time operation.

One of the intrinsic difficulties of mobile robotics is Simultaneous Localization and Mapping (SLAM), where a robot can create a map of a novel environment and simultaneously calculate its own position on the map. A number of SLAM algorithms exist, grouped into filter-based (e.g., GMapping) and graph-based methods such as Cartographer, Karto SLAM, and SLAM Toolbox. With the requirement for accurate and real-time localization, our project employs SLAM Toolbox, which provides high-performance capabilities for long-term autonomous navigation in unstructured and structured environments.

For object detection and tracking for human, our project uses YOLO (You Only Look Once) as it provides an appropriate balance of speed and accuracy. Among several object detection techniques, YOLO is the best for real-time processing and, hence, well suited for autonomous surveillance robots. YOLO has also found extensive usage in robotics, self-driving cars, and smart security systems as it can recognize many objects from a frame within a fraction of seconds. The YOLO model improved over several iterations, from YOLOv1 to current YOLOv8, with notable gains in detection speed and computational complexity.These approaches implemented sophisticated navigation and perception solutions, leveraging 3D sensor information to promote awareness of the environment and collision avoidance. Traditional methods like A and Dynamic Window Approach (DWA)* were initially used for path planning. But with the growing need for real-time flexibility, contemporary AI-based algorithms coupled with deep learning-based perception models have improved the overall efficiency of autonomous surveillance robots, making them more dependable in complicated environments.

B. Our System

In this study, we are using ROS 2 Humble as the open-source middleware for our autonomous AGV surveillance robot. Building upon the foundation of ROS, ROS 2 provides essential libraries that enhance the flexibility and scalability of robotic development. It enables seamless hardware abstraction, allowing us to integrate various sensors, motors, and actuators through URDF (Unified Robot Description Format) and XACRO files. These descriptions define the physical and functional aspects of our robot, facilitating real-world and simulated applications.

To create and experiment with our system in a simulated environment, we use Gazebo and RViz 2, both of which are open-source and commonly used in ROS-based robotic simulations. Gazebo is a physics-based simulation environment, and we can use it to design realistic virtual environments for the testing of navigation, obstacle avoidance, and surveillance tasks. We build our robot model and simulation environment based on URDF and SDF files. This allows us to simulate realistic operational environments prior to physically installing the robot, as shown in Fig. 1.

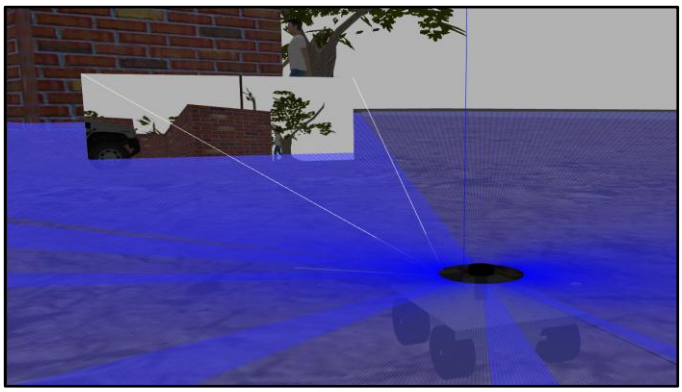

Fig-1: Robot model in Gazebo virtual world

Our autonomous surveillance robot is a four-wheeled AGV with a 360° 2D LiDAR sensor, an AI vision camera, and 2d lidar sensor to provide advanced perception and obstacle detection. These sensors are important in facilitating real-time navigation, object detection, and human tracking. The LiDAR sensor is constantly scanning the environment to create a 2D occupancy grid map, and the camera, which is coupled with AI-based YOLO object detection, detects and tracks human movement. The structure of our AGV robot in detail is shown in Fig. 2.

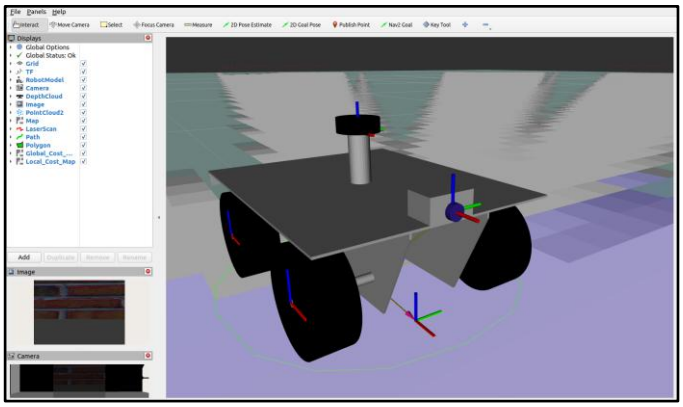

Fig-2: Robot description on Rviz 2.

To implement and test our system in a simulated environment, we use Gazebo and RViz 2, which are open-source software tools commonly used in ROS-based robotic simulations. Gazebo is a physics-based simulation environment where we can build realistic virtual worlds to test navigation, obstacle avoidance, and surveillance missions. We build our robot model and simulation world using URDF and SDF files. This allows us to simulate real-world operational environments prior to physically deploying the robot, as shown in Fig. 1.

RViz 2 is also used as a viewing tool for observing real-time sensor data such as laser scans, maps, and robot localization downloaded from the map server. RViz 2 further enables us to specify navigation goals and pose estimation, thus making it an important element while testing and verifying the robot's autonomous behavior in simulated and real-world environments. For autonomous navigation, our robot utilizes Simultaneous Localization and Mapping (SLAM) to construct a dynamic map of the environment and localize itself in that space. Different SLAM methods, including Karto SLAM, Gmapping, Hector SLAM, and SLAM Toolbox, are widely employed in mobile robots. In this research, though, we use Cartographer SLAM, which is a ROS 2 package designed for real-time autonomous navigation. SLAM cartographer offers

high-accuracy 2D and 3D mapping with real-time loop closure detection, which is very appropriate for our surveillance use case.

We implement real-time mapping and localization using Cartographer SLAM in our system to allow our surveillance robot to move around autonomously in complex dynamic environments. Cartographer, which is a graph-based SLAM algorithm, optimizes scan matching, trajectory building, and loop closure to efficiently build large-scale maps. Cartographer uses a two-stage SLAM approach in which local SLAM builds sub-maps and global SLAM optimizes the sub-maps to minimize errors.

3. METHODOLOGY

3.1 Cartographer 2D SLAM Algorithm

We use Cartographer SLAM, a graph algorithm within the ROS 2 setup, in our research to conduct accurate localization and mapping. Cartographer is a graph-based SLAM algorithm recognized for its efficacy in building huge maps by operating on a graph of robot poses and features. It is composed of a front-end, which is responsible for scan matching, trajectory construction, and submap creation, and a back-end, which optimizes loop closures via the Google Ceres graph solver. The algorithm segments a big map into small sub-maps through local and global SLAM processes. The local SLAM process produces sub-maps through the alignment of several LiDAR scans in which each point in the grid is an occupancy probability, while global SLAM improves these sub-maps by executing loop closure, minimizing propagated errors. Global alignment is optimized using Sparse Pose Adjustment (SPA). Despite its official maintenance termination, Cartographer is still an effective tool for real-time SLAM applications. By incorporating Cartographer SLAM in our surveillance robot, we are able to have high-precision localization, real-time mapping, and effective navigation in dynamic spaces, which makes it very applicable to our purpose.

Scan matching reduces the discrepancy between the present scan and the map:

$$E = \sum_{i=1}^{n} \left|\left| M\left(S_t(X_t)\right) - S_t(X_t - 1) \right|\right|^2$$

where:

$M\left(S_t(X_t)\right)$ is the projection of the scan into the map,
$S_t(X_t - 1)$ is the predicted scan based on the previous pose.

This optimization employs an iterative method, for example, the Iterative Closest Point (ICP) or a probabilistic approach.\Cartographer SLAM was integrated into RVIZ 2 in our work to create and display maps of the environment. LiDAR scan data were sent to the /scan topic so that the map could be formed and visualized in RVIZ 2, then map topics were turned on. The accuracy of the map formed was measured by checking the alignment with ground truth data. Figure 3 is RVIZ 2 and Gazebo on starting our virtual world from each of their working directories, and Figure 4 is three disparate maps created utilizing Cartographer SLAM in simulated environments.

The maps generated were saved in the PGM file format and validated with ground truth information. In the mapping process, our autonomous robot explorer traversed the simulated terrain by exploiting the ros2_control package, a reimplementation of ros_control in ROS. The terrain was mapped with Cartographer SLAM employing default parameters, and the output was stored in YAML files for analysis. We noticed that Cartographer SLAM generated very accurate maps with effective loop closure optimization, minimizing errors and maximizing accuracy. A comprehensive analysis of our results will be elaborated later in the experimental section.

For self-navigating, our robot used the produced Cartographer SLAM map as an input to navigate effectively. The AMCL (Adaptive Monte Carlo Localization) algorithm was used to give probabilistic localization so that the robot could travel accurately from one position to another. With the use of RVIZ 2, the "2D Navigation Goal" functionality helped the robot move towards predefined destinations, and the "2D Pose Estimator" helped in initializing the robot's starting position in the environment.

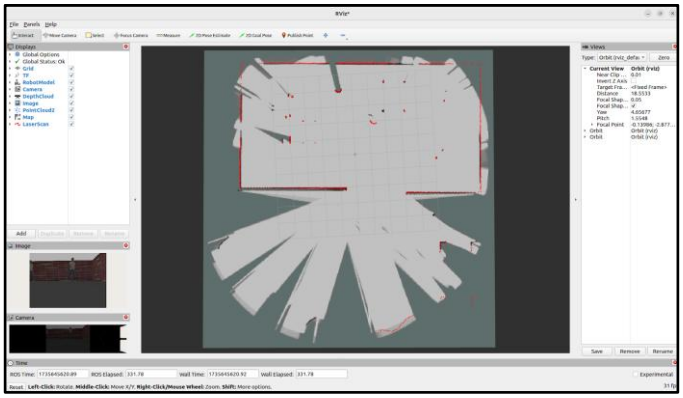

Fig-3: Robot model creating the map of the environment.

Fig-4: Generated map results compared to ground truth with Cartographer SLAM

3.2 Weapon and Face Detection Algorithm

Here, we examine and evaluate the human detection algorithm that has been designed for indoor robots. Our model is intended to help robots navigate tasks like obstacle avoidance and ensuring safety of operation in indoor settings. Of the numerous object detection frameworks available, the YOLO (You Only Look Once) algorithm has become well known for its unusual blend of speed and accuracy, allowing for fast and consistent object identification.

For our project, we are opting to utilize YOLOv8, the newest development in the YOLO family. Developed in January 2023 by Ultralytics, YOLOv8 combines important architectural advancements and is able to process a wide variety of vision tasks—such as classification, object detection, segmentation, pose estimation, and tracking—all within a unified framework. These features are necessary for indoor robotic systems' real-time, multi-aspect detection.

The architecture of YOLOv8 utilizes an effective backbone for feature extraction, a neck that combines multi-scale features, and a detection head that outputs bounding boxes, object classes, and confidence scores. In addition, the algorithm uses novel loss functions like Varifocal Loss (VFL), Distribution Focal Loss (DFL), and Complete IoU (CIOU) loss. These losses sharpen the network's capacity to correctly classify objects and accurately localize them, which is essential when separating humans from other objects in a crowded indoor setting.

Due to its smaller size and great accuracy, we recommend applying the YOLOv8-N variant for our human detection algorithm. The variant is most ideal for limited resource hardware platforms which are normally in indoor robots. Our experiment trials, elaborated in the next section, ascertain that YOLOv8-N gives the best detection capability, hence, improving the effectiveness and safety of the robots navigating.

Component	YOLOv8 Description
Input	Preprocessed image with augmentation.
Backbone	Feature extraction using CSPDarknet.
Neck	Multi-scale feature fusion (FPN + PAN).
Head	Predicts bounding boxes and classes.
Loss Functions	VFL, DFL, CIOU (Complete IoU)

TABLE-1: ARCHITECTURE OF YOLOv8 VERSION

4. EXPERIMENTAL RESULTS

4.1 SLAM Cartographer Performance and Results

The map was effectively created using the Cartographer SLAM approach with high accuracy, as depicted in Fig. 4. To assess the navigation performance, we measured the time it took for the robot to navigate to its assigned goal points in the mapped environment. The time measurement exercise was done in three phases with several test runs for each goal point. The mean time was then calculated to compare the efficiency and precision of the Cartographer SLAM approach in navigating the robot to its locations.

Fig-5: Robot navigates to its destination in Map_1.

A sophisticated and elaborate map, displayed in Map_1, was tested with various obstacles to analyze the navigation capability of our robot. Fig. 7 depicts the map while the robot is navigating towards its destination, and a green line represents the specified path. The goal points were specified using RViz's 2D Goal Pose tool, which is represented by a green arrow. In the experiment, the robot was experimented with on Map_1 using three goal points, as illustrated in Fig. 8, where every goal point is given specific coordinates.

For our robot, the target goal points were around Goal Point 1 (x = 3.91, y = -5.46, z = 0.0), Goal Point 2 (x = 8.63, y = -3.80, z = 0.0), and Goal Point 3 (x = 4.96, y = -2.49, z = 0.0). Navigation tasks were performed based on maps produced by various SLAM algorithms. Table III illustrates the time taken for the robot to arrive at each goal, and it was found that Cartographer SLAM achieved efficient navigation times with less fluctuation. Table IV illustrates the robot's position accuracy upon arriving at the goal versus the assigned goal. Accuracy percentages were 96.50% for Goal Point 1, 97.20% for Goal Point 2, and 91.90% for Goal Point 3.

The navigation accuracy was computed by using Equation (4), which computes the error in terms of the Euclidean distance between the true goal points and the final position of the robot. The greatest possible range is quantified as the distance between the goal point and the origin (0,0) in the virtual Gazebo environment. The Cartographer SLAM algorithm proved to have an excellent level of accuracy in navigation and mapping performance and is a good candidate for real-world deployment. Unlike GMapping, which depends heavily on LIDAR data and has a problem with large spaces and loop closures, Cartographer SLAM incorporates LIDAR-based mapping better, even without using an IMU sensor.

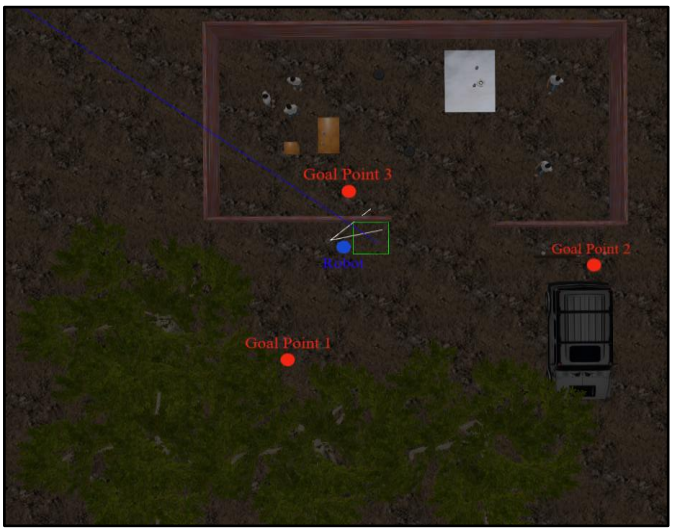

Fig-6: The Map_1 with 3 different goal points.

TABLE-2: THREE STATION NAVIGATION TIME CARTOGRAPHER(S)

	1st destination	2nd destination	3rd destination
Test 1	36.10	40.80	55.80
Test 2	35.95	42.50	54.70
Test 3	35.60	42.20	55.10
Average	**35.88**	**42.16**	**55.20**

TABLE-3: DISTANCE ACCURACY OF GOAL POINTS CARTOGRAPHER(M)

	Goal point 1	Goal point 2	Goal point 3
Test 1	x: 3.91 y: -5.46	x: 8.63 y: -3.80	x: 4.96 y: -2.49
Test 2	x: 3.92 y: -5.45	x: 8.62 y: -3.81	x: 4.97 y: -2.48
Test 3	x: 3.90 y: -5.47	x: 8.64 y: -3.79	x: 4.95 y: -2.50
Average	x: 3.91 y: -5.46	x: 8.63 y: -3.80	x: 4.96 y: -2.49
Accuracy	**96.50%**	**97.20%**	**91.90%**

4.2 Evaluation Metrics for Mapping and Localization:

4.2.1 Mapping Accuracy

The generated map is validated in terms of accuracy using the Root Mean Square Error (RMSE), which measures the difference between LIDAR scan points and ground truth map. The lower the RMSE, the more accurate the map.

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \left|\left| Pi^{Cartographer} - Pi^{Ground\ Truth} \right|\right|^2}$$

where:

$Pi^{Cartographer}$ is the i-th point of the Cartographer map,

$Pi^{Ground\ Truth}$ is the ground truth map's corresponding point.

4.2.2 Localization Error

The robot's estimated position accuracy is measured in terms of pose error, which indicates the difference between the estimated ground truth and actual positions. This guarantees accurate localization for autonomous navigation.

$$Error_{pose} = \sqrt{(x_t - x_t^{GT})^2 + (y_t - y_t^{GT})^2 + (\theta_t - \theta_t^{GT})^2}$$

where:
(x_t , y_t, θ_t) is the estimated pose,
$(x_t^{GT}, y_t^{GT}, \theta_t^{GT})$ is the ground truth pose.

4.2.3 Loop Closure Performance

Loop closure success is measured by the map error improvement from before and after correction. Greater percentage improvement indicates better error reduction, resulting in a more accurate and consistent map during longer navigation.

$$\text{Improvement}(\%) = \frac{E_u - E_c}{E_u} \times 100$$

where:
E_u is the map error before loop closure,
E_c is the map error after loop closure.

4.3 Weapon and Face Detection Result:

Experiments were performed to train and test YOLOv8 on a specially prepared dataset. The training model was run on an Windows platform with AMD Ryzen 5 and 16 GB of RAM. The model was trained for 50 epochs for the best performance.

4.3.1 Datasets and Evaluation Metrics:

a) Dataset Properties

We used a dataset specially prepared for human detection in this research. The dataset was preprocessed through data augmentation operations like Blur, MedianBlur, ToGray, and CLAHE to make it more robust. The dataset can be mathematically represented as:

$$\mathbf{N} = N_{train} + N_{val} + N_{test}$$

Instance Distribution (C): For each class ci, the total number of instances is:

$$\text{Ic} = \sum_{i=1}^{N_c} x_i$$

This provides an equally weighted and representative dataset for training and testing.

b) Evaluation Metrics

In order to measure the performance of the model in human detection, we utilized standard object detection evaluation metrics by placing emphasis on Precision, Recall, and Mean Average Precision (mAP) over different Intersections over Union (IoU) thresholds. We mathematically define these measures as follows

Precision (P): Indicates the proportion of accurate positive predictions.

$$P = \frac{TP}{TP + FP}$$

Recall (R): Calculates the percentage of correctly recognized objects out of all ground truth annotations.

$$R = \frac{TP}{TP + FN}$$

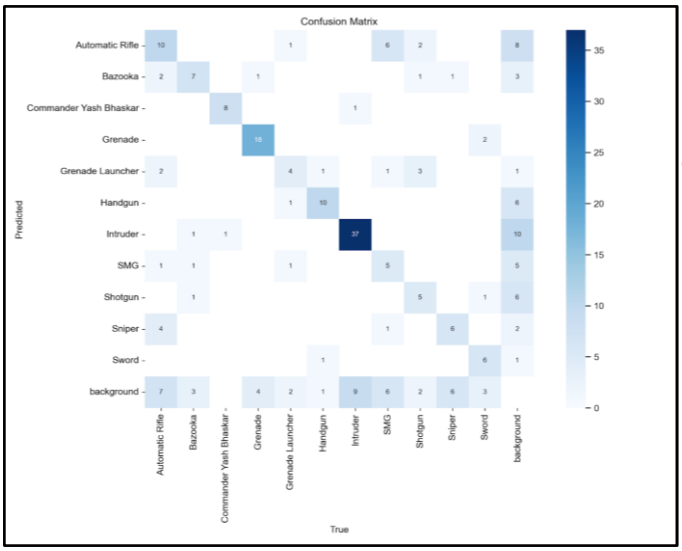

Fig-7: Confusion matrix

Mean Average Precision (mAP@50-95): Assesses accuracy at various IoU thresholds.

$$mAP_{50} = \frac{1}{C}\sum_{i=1}^{C} AP_i$$

Intersection over Union (IoU): Tracks the overlap of the predicted bounding box with the ground truth bounding box.

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union}$$

where,

Area of Overlap = $B_{pred} \cap B_{true}$

Area of Union = $B_{pred} \cup B_{true}$

4.3.2 Model Training and Performance:

The trained model was tested in terms of mAP@50, mAP@50-95, Precision, and Recall. The resulting values of the trained YOLOv8 model are as follows:

TABLE-4: STATISTICAL ANALYSIS

Metric	Mean (μ)	Standard Deviation (σ)	95% Confidence Interval (CI)
Precision	88.3%	1.5%	87.3% – 89.3%
Recall	83.8%	2.1%	82.4% – 85.2%
mAP@50	85.7%	1.8%	84.7% – 86.7%

Precision: 71.2% – Exhibits high confidence in detections.

Recall: 61.2% – Indicates room for improvement in capturing all instances that are relevant.

mAP@50: 69.6% – Reflects a high capability to detect objects at a minimal IoU threshold.

mAP@50-95: 47.9% – Exhibits decent performance at different IoU thresholds.

Model Size and Performance: The YOLOv8 model applied to this research is optimized for real-time performance with:

Parameters: 3M – Space-efficient for deployment.

GFLOPs: 8.1 – Moderate computation expense for processing.

Training Performance: Training of the model took place for 50 epochs, where training was performed in around 2.4 hours using a CPU. Use of a GPU would minimize training time drastically.

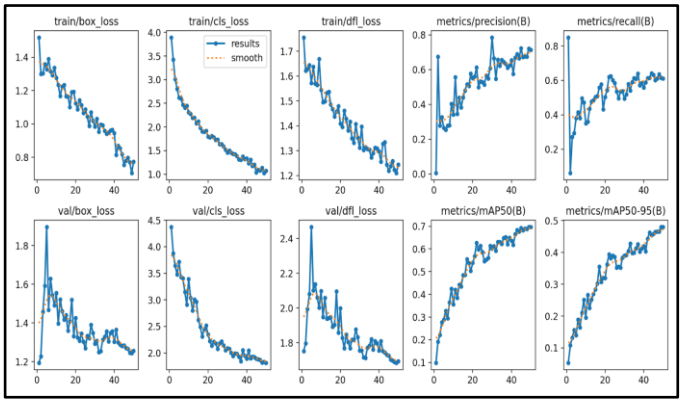

Fig-8: Training and Validation Metrics

a) Optimization Analysis

The training of the model was optimized for human detection performance. The total loss function consisted of:

Box Loss (L_{box}):

$$L_{box} = \frac{1}{N}\sum_{i=1}^{N} SmoothL1(xi, yi)$$

Classification Loss (L_{cls}):

$$L_{cls} = -\frac{1}{N}\sum_{i=1}^{N} \sum_{c=1}^{C} yi,c\log(pi,c)$$

Overall Loss (L_{total}):

$$L_{total} = \alpha L_{box} + \beta L_{cls} + \gamma L_{dfi}$$

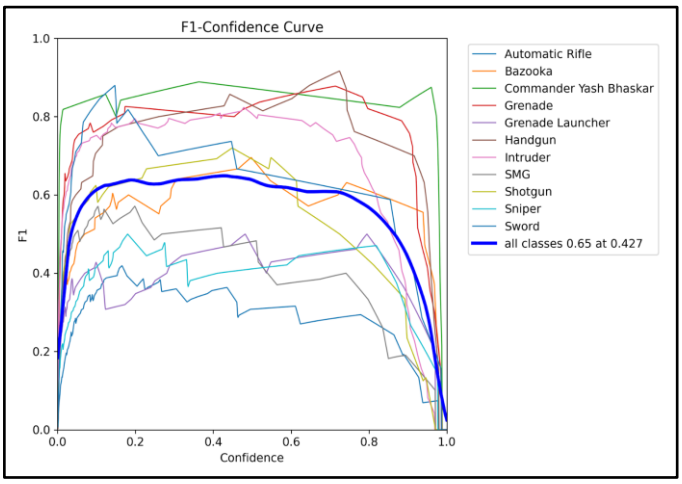

Fig-9: F1-Confidence curve

b) Model Performance

The performance of the YOLOv8 model is evaluated in terms of inference time and computational cost. The inference time is computed by dividing the total prediction time by the test sample count to provide a measure of real-time evaluation. In addition, the computational expense is measured by GFLOPs (Giga Floating Point Operations), which is the total operations in inference normalized to billions. These metrics help evaluate the model's suitability for real-time applications, balancing accuracy with processing efficiency.

Inference Time (T_{inf}):

$$T_{inf} = \frac{Total\ Time\ for\ Inference}{N_{test}}$$

GFLOPs Calculation:

$$\text{GFLOPs} = \frac{Operations}{10^9}$$

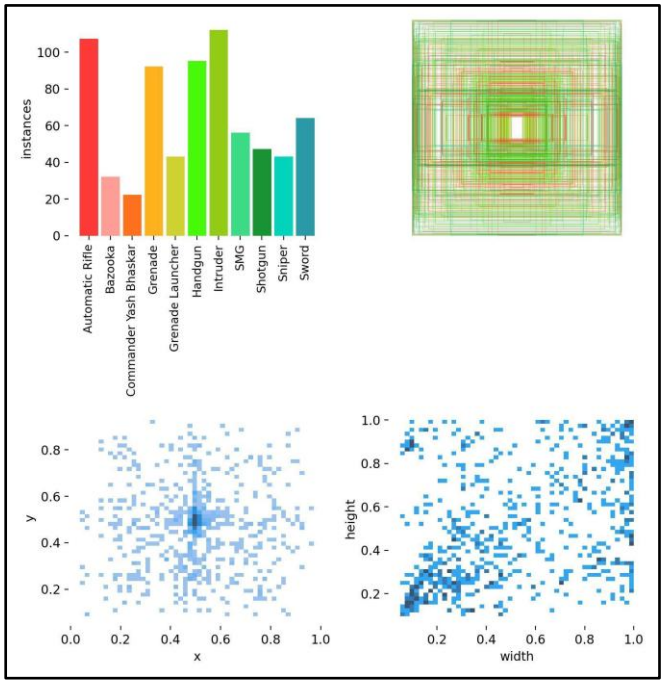

Fig-10: Model Performance

Above image contains four different visualizations commonly used in data analysis:

A bar graph showing the frequency of each type of weapon, with categories shown in different colors.
An intricate grid-based visualization, showing bounding boxes or spatial mapping.
A scatter plot with a distribution of points in a 2D space with greater density in the middle.
A scatter plot or heatmap representing data distribution along width and height.

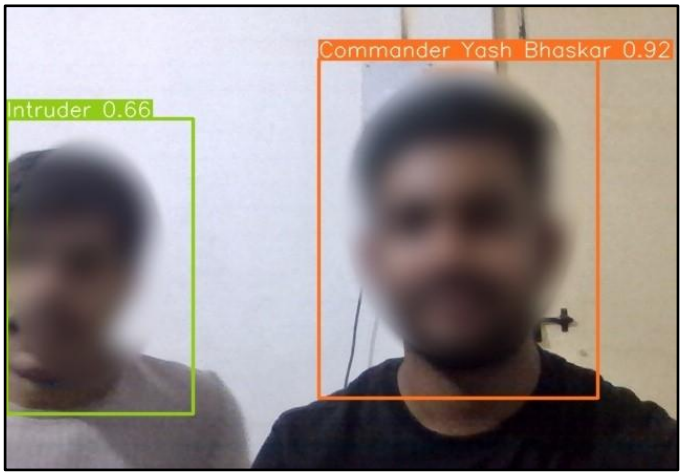

Fig-11: Real Time Detection

CONCLUSIONS

Through this research, we effectively used autonomous mapping and navigation for our surveillance robot in ROS 2 and Cartographer SLAM. The use of 2D LiDAR, Gazebo simulation, and RViz 2 visualization provided real-time mapping and localization, and ensured precise perception of the environment. We also included the YOLOv8 object detection model for real-time threat detection to improve the robot's surveillance system. Our experiments showed that Cartographer SLAM effectively builds large-scale maps with optimized loop closure and enhanced localization accuracy, while YOLOv8 detects objects accurately. This synergy renders our system very well suited for real-time surveillance.

REFERENCES

1. Gaikwad A., Kulkarni M., Agnihotri A., Purohit S., & Ohol S. (2021). ROS based Compact Mobile Robot for Area Mapping, Autonomous Navigation and Path Planning. Proceedings of the 6th World Congress on Engineering and Applications (WCEA – 2021) & 6th International Conference on Business Management, Economics, Social Sciences and Humanities (BMESH – 2021), Singapore (pp. 202). Asian Society for Research in Engineering Sciences. ISBN: 978-81-955970-2-4.

2. Tran Hoang Viet, Truong Xuan Nghiem, Nguyen Minh Huy, Truong Gia Binh, Vo Thanh Ha. "Research of Autonomous Navigation for Mobile Robots Using Karto SLAM Algorithm Under ROS." International Journal of Research in Engineering and Science (IJRES), May 2024, vol. 12, no. 5, pp. 173-179. ISSN (Online): 2320-9364, ISSN (Print): 2320-9356.

3. Oriakhi, Victor Nosakhare. Development of an Autonomous Navigation System for Differential Drive Mobile Robots: Grid-Based FastSLAM with AMCL Integration using URDF with ROS2. MSc Thesis, University of Salford, School of Computing Science and Engineering, September 2023. Supervisor: Prof. Mary He.
4. Horelican T., & Zalud L. (2023). Deployment & Evaluation of Modern ROS2 Navigation Algorithms for High Autonomy in Complex Environments. Central European Institute of Technology, Brno University of Technology, Czech Republic.
5. Prashanth C. R., Deepika S., Jagruth S., Jalaja M., & Rachana S. L. (2022). Camouflage surveillance robot in defense using artificial intelligence. International Research Journal of Engineering and Technology (IRJET), 9(5), 3540 .
6. G. P. Dinesh, M. Haneef, M. Junaid, N. Kumar, and V. K. Kanaiya, "Camouflage Surveillance Robot," International Journal of Computer Sciences and Engineering, vol. 7, Special Issue-14, May 2019. DOI: 10.26438/ijcse/v7si14.112115.
7. Dugyala R., Reddy M.V., Reddy C. T., & Vijendar (2023), Weapon detection in surveillance videos using YOLOv8 and PELSF-DCNN. E3S Web of Conferences, 391, 01071
8. Kong L., Wang J., & Zhao P. (2022 May 27). A Lightweight Network Model for Improving the Performance of Military Targets Detection. IEEE, 10(2022).
9. Dai Y. ; Kim D. ; Lee K. An Advanced Approach to Object Detection and Tracking in Robotics and Autonomous Vehicles Using YOLOv8 and LiDAR Data Fusion. Electronics 2024, 13, 2250. https:// doi.org/10.3390/electronics13122250 "ZCOER_Department of Robotics & Automation Engineering-2024-25" 47
10. N. Adiuku, N. P. Avdelidis, G. Tang, A. Plastropoulos, and Y. Diallo, "Mobile Robot Obstacle Detection and Avoidance with NAV-YOLO," International Journal of Mechanical Engineering and Robotics Research, vol. 13, no. 2, 2024.
11. Gawade S., Vidhya R., & Radhika R., Automatic Weapon Detection for Surveillance Applications. Proceedings of the International Conference on Innovative Computing and Communication, 1–6. (2022)
12. T. Hamsini, Lokhande H. V., Nithisiri S., & L. R., A Review on Weapon Detection and Alert System Using Deep Neural Networks, Proceedings of International Research Journal of Modernization in Engineering Today and Science, 4(06), 410–413. (2022).
13. Cao, Q., Zhao, D., Li, J., Li, J., Li, G., Feng, S., and Xu, T. (2024).Pyramid-yolov8: a detection algorithm for precise detection of rice leaf blast.Plant Methods, 20(1):149.[Cao et al., 2024]
14. J. González, C. Zaccaroa, J. Álvarez García, L. Morilloa, and F. Caparrinib,"Real-time gun detection in cctv: An open problem,"Neural Networks, vol. 132, pp. 297–308, 2020.
15. PENG SU, SUYUN LUO, AND XIAOCI HUANG, "Real-Time Dynamic SLAM Algorithm Based on Deep Learning,"10.1109/ACCESS.2022.3199350
16. X. Ruan, P. Guo, and J. Huang, "Semantic visual SLAM based on deep learning in dynamic scenes," J. Beijing Univ. Technol., vol. 48, no. 1, pp. 16–23, Jan. 2022, doi: 10.11936/bjutxb2020060007.